基于词向量与 TextRank 的关键词提取方法 *

周锦章，崔晓晖†

(武汉大学 国际软件学院，武汉 430072)

摘　要：针对词汇语义的差异性对 TextRank 算法的影响进行了研究，提出一种基于词向量与 TextRank 的关键词抽取方法。首先，利用 FastText 将文档集进行词向量表征；其次，基于隐含主题分布思想和利用词汇间语义性的差异，构建 TextRank 的转移概率矩阵；最后，进行词图的迭代计算和关键词抽取。实验结果表明，该方法的抽取效果相比于传统方法有明显提升，同时证明利用词向量能简单而有效的改善 TextRank 算法的性能。

关键词：关键词抽取；语义差异性；TextRank；词向量；隐含主题分布

中图分类号：TP　　doi: 10.3969/j.issn.1001-3695.2017.11.0787

Keyword extraction method based on word vector and textrank

Zhou Jinzhang, Cui Xiaohui†

(*International School of Software*, *WuHan University*, *Wuhan* 430072, *China*)

Abstract: The influence of lexical semantic difference on TextRank algorithm is studied, this paper presents a keyword extraction method based on word vector and TextRank. Firstly, it used FastText to represent word vector from the document corpus. Then, based on the idea of implicit subject distribution and used the differences in lexical semantics to build a probability transfer matrix for TextRank. Finally, iterative calculate the lexical graph model and extracted keywords. Experimental results show that the extraction performance of this method is significantly improved compared with the traditional method. In addition, it is proved that the use of word vectors can improve the performance of TextRank algorithm simply and effectively.

Key words: keyword extraction; semantic difference; TextRank; word vector; implied subject distribution

0　引言

关键词抽取在文本处理的许多领域中是一项重要技术，如：文本聚类、文本摘要和信息检索。在当下大数据时代，关键词抽取更是在 NLP 领域扮演着重要角色，为情感分析、语义分析、知识图谱等热点问题提供了基石。目前该领域主流代表的方法有基于隐含主题模型的关键词抽取（LDA[1]）、基于 TF-IDF[2]词频统计的关键词抽取和基于词图模型的关键词抽取（TextRank[3]）。

以上的三种算法因其简洁而有效，所以被广泛运用。为了进一步提升抽取效果，刘俊等人[4]利用主题模型中词和主题的分布情况计算词的主题特征，并将该特征与关键词抽取中的常用特征结合，用装袋决策树方法，构造一个关键词抽取模型；罗燕等[5]利用词频统计规律改进传统的 TF-IDF 算法，提升了关键词抽取效果；耿焕同等人[6]在词频统计的基础上结合词共现图来找出频率较低的主题词来提升结果；顾益军和夏天[7,8]分别利用 LDA 和词向量聚类结合 TextRank 进行关键词抽取；李鹏等人[9]用 Tag 值改进文档图节点的边权值的计算，并且将不同 Tag 值结果融合，提出了新的 Tag-TextRank 算法；李跃鹏等人[10]利用 word2vec 训练得到的词向量，计算词语的相似度然后通过词语聚类进行关键词抽取；姜芳等人[11]通过计算词语的语义距离对词语进行密度聚类，得到主题相关类，然后从中选取中心词作为关键词；Ortega 和 Fermín 等人[12,13]利用标记过的语料库将 TextRank 算法从无监督变为有监督算法，从而达到提升效果的目的。

上述研究将三种主流算法单独或者组合改进达到提升效果的目的，但是在这三种主流算法中效果较为明显且不依赖其他文档的是 TextRank 算法，这也是该算法最大的优点。该算法从 PageRank 算法[14]得到启发而来，TextRank 算法是用于文本的基于图的排序算法，通过将文本切分成单独的词语，通过词共现关系建立词图模型，利用投票原理将文本中的重要词语进行排序，最终达到抽取关键词的目的。关键词是来源于当前文档且能描述文档主题的一系列词语，所以仅仅是考虑词语的位置关系是不够的，本文从隐含主题分布思想得到启发：一篇文档包含有多个隐含主题，每个隐含主题下又包含有多个文档中的词语，不同主题之间的词语具有明显的语义差异性，而一篇文

收稿日期：2017-11-27；**修回日期：**2018-01-10　　**基金项目：**中央高校基本科研业务费专项资金（2042017gf0035）

作者简介：周锦章（1993-），男，湖北黄冈人，硕士研究生，主要研究方向为数据挖掘、机器学习等；崔晓晖（1971-），男（通信作者），教授，博导，博士，主要研究方向为大数据安全等（xcui@whu.edu.cn）.

档具有多个隐含主题，关键词也来源于这些隐含主题，所以本文提出以下方法：首先利用 FastText 工具来训练数据，获得词向量，利用词向量计算词汇间的语义性差异来改进 TextRank 中词语的转移概率矩阵，让权重更多的转移给语义性差异更大的词语，从而能增加从不同隐含主题中抽取到关键词的概率，最终提升关键词抽取效果。通过实验证明了本文所提出方法的可行性，简单而有效的提升了原有算法的效果。

1 方法原理及流程

一篇文章往往包含着不同的主题，而关键词也是来源于这些不同的主题，从理论和实际来看，这些关键词从语义的角度分析大部分的语义差异性很明显，所以这是一个特性。在 TextRank 算法中，文档中的词语是通过共现关系来构建图模型，通过平均转移概率矩阵进行迭代计算每个词语权重，最终收敛后，将权重进行排序，选择 Top*K* 个词语作为关键词。这样的做法很容易将在文档中出现频率高的词语抽取出来，但是一篇文档的关键词不仅仅是出现频率高的，而且有时候出现频率高的词语却不一定是关键词。语言文字是高度抽象的符号，所以从语义角度分析一篇文档的关键词很重要。综合以上论述，本文就从隐含主题分布思想和语义差异性角度提出了基于词向量和 TextRank 的关键词提取方法。

方法的流程分为两步：a)利用 FastText 工具对文档数据集进行训练，得到词向量表征；b)计算当前文档中各个词语的角余弦位距，也就是对词语的语义差异性进行量化，用该结果对原始 TextRank 算法的权重转移概率矩阵进行改进，迭代计算至收敛，提取 Top*K* 个词语作为关键词。方法的流程如图 1 所示。

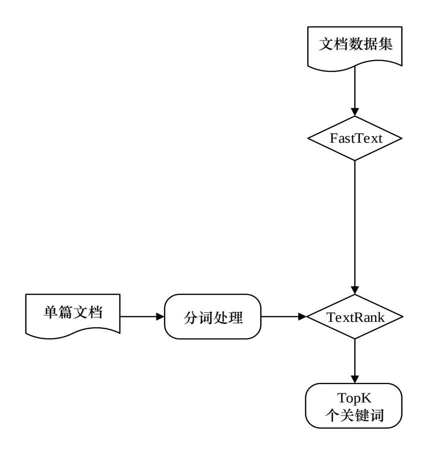


图 1 关键词抽取流程示意图

2 FastText 生成词向量

FastText 是 Facebook 开发的一款快速文本分类工具，简洁而高效的解决了文本分类和表征学习的问题。实际上，该项目分为两个部分，相关内容为文献[15,16]，在此，本文主要利用文献[15]的相关研究结论。

词向量是使用向量来表达词语，这类方法中目前较为出名的是 2013 年 Mikolov 等人提出的 Word2vec，它基于浅层神经网络训练语料，将词语嵌入到相应维度的空间中，得到的结果就是词向量。利用 FastText 工具生成词向量是基于 CBOW(Continuous Bag-of-Words)模型和 Skip-gram 模型。

CBOW 是根据上下文的词语预测当前词语出现概率的模型。如图 2 所示，该模型总共分为三层：输入层、投影层和输出层。

a)输入层即为当前单词周围的 n 个单词的词向量，记当前词语为 $w(t)$，则周围的 n 个词语可以记为…$w(t-2)$，$w(t-1)$，$w(t)$，$w(t+1)$，$w(t+2)$ …，那么这些词的编码表示为…$V(w(t-2))$，$V(w(t-1))$，$V(w(t))$，$V(w(t+1))$，$V(w(t+2))$ …。从训练文档中抽取 N 个不重复的词语组成词汇表，对该词汇表的所有词语进行 one-hot 编码，这就是将输入层词语编码的过程。

b)投影层即将输入层的所有词语的编码进行求和操作。

c)输出层即将语料中的全部词语作为叶子节点，词频作为节点的权，构建 Huffman 树。

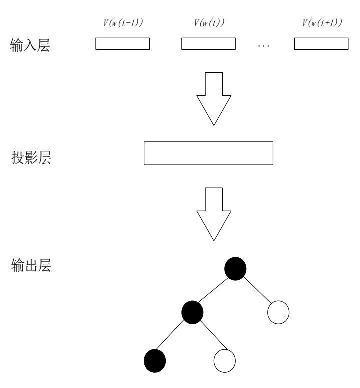


图 2 CBOW 模型示意图

Skip-gram 模型的原理 CBOW 模型正好相反，是通过当前词语预测上下文。Skip-gram 模型同样分为三层：输入层，投影层，输出层。如图 3 所示。

a)输入层是当前词语的 one-hot 编码。

b)投影层是将输入层的词语编码和权重矩阵进行索引计算，得到当前词语的词向量。

c)输出层是一个 softmax 分类回归器，每个节点会输出 0-1 之间的概率值，这些概率值之和为 1。

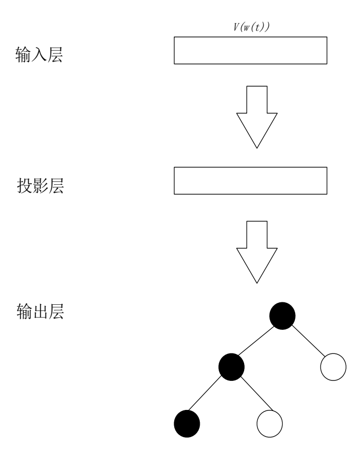


图 3 Skip-gram 模型示意图

3 利用词向量改进词节点权值

将一篇文档转换成词图模型，是将文档中的每个词看做一个节点，每个节点之间的边由词节点之间的词共现关系决定，而节点的重要性又由相邻节点指向数量决定。TextRank 算法的原始数学表示如式(1)所示。

$$WS(V_i)=(1-d)+d*\Sigma_{v_j\in In(v_i)}\frac{w_{ji}}{\Sigma_{v_k\in Out(v_j)}w_{jk}}WS(V_j) \quad (1)$$

构建关键词图 $G=(V,E)$，其中 V 为节点集合，E 为节点之间的边集合。$In(v_i)$ 是指向节点 V_i 的节点，$Out(v_j)$ 是节点 V_j 指向的节点，w_{ji}、w_{jk} 是两节点之间的边权，$WS(V_i)$ 是 V_i 节点的权重，d 是阻尼系数，一般取值为 0.85，其意义是当前节点向其他任意节点跳转的概率，同时能够让权重能够稳定的传递至收敛。

在利用 TextRank 算法进行关键词提取的主要步骤如下：

a)将当前文本进行整句分割，得到 $\mathrm{T}=[s_1,s_2,\cdots,s_n]$；

b)对于 $s_i\in T$,进行分词、词性标注、停用词过滤，必要时可添加特定停用词词典，最后得到 s_i=$[\mathrm{d}_{i1},\mathrm{d}_{i2},\cdots,\mathrm{d}_{in}]$，$d_{in}\in s_i$ 为处理后的候选关键词；

c)构建关键词词图 $G=(V,E)$，其中 V 为候选关键词节点集合，E 为候选关键词之间的边集合，边的有无由候选关键词的共现关系决定，共现则有边，否则无；

d)根据上面公式，迭代传播候选关键词节点 V_i 的权重，直至收敛；

e）得到所有候选关键词节点 V_i 的权重，进行降序排列，得到 Top*K* 个词作为最终关键词。

上述过程为一般 TextRank 算法提取关键词过程。在 TextRank 算法中，需要迭代计算候选关键词的权重直至收敛，该过程被称为马尔可夫过程，它的数学解释是：在已知目前状态的条件下，它未来的演变不依赖于它以往的演变，所以迭代结果将与候选关键词的初始权重以及边的权重无关，而将只与候选关键词权重转移概率矩阵有关。在 TextRank 算法中，候选关键词之间的边由共现关系决定，而对共现关系有重要影响的参数是共现窗口 window，其大小为 w，表示每次最多出现 w 个词语，通过每次向右滑动一个窗口来建立词语间的共现关系，最终以此来构建权重转移概率矩阵。w 的大小需要通过实验取得，过小会导致权重转移概率矩阵稀疏，过大会导致权重转移概率矩阵稠密，两种情况均会导致抽取结果误差较大。

同时根据文献[1]，边权的值对收敛结果没有影响，所以在本文实验中，所有具有共现关系的顶点之间的边权值设为 1，每个顶点的初始权重设为 $\frac{1}{n}$（n 为顶点个数）。则上式转换为

$$\mathrm{WS}(V_i)=(1-d)+d*\sum_{v_j\in In(v_i)}\frac{1}{|Out(V_j)|}WS(V_j) \quad (2)$$

在实际运算过程中采用的是矩阵运算，则上式转换为

$$\mathrm{WS}(V_i)=(1-d)+d*E*WS(V_j) \quad (3)$$

根据式（3），每一个顶点的权值在迭代过程中将是均匀的转移给每一个与其相连的顶点，初始权重转移概率矩阵 E 如式（4）所示。

$$E=\begin{pmatrix} e_{11} & \cdots & e_{1n} \\ \vdots & \ddots & \vdots \\ e_{n1} & \cdots & e_{nn} \end{pmatrix} \quad (4)$$

初始权重转移概率矩阵 E 中元素的值由顶点之间的边决定，有则为 1，无则为 0，$e_{nn}=[0,1], e_{nn}\in N$。根据本文的设想及验证，本文提出利用候选关键词的角余弦位距构建权重转移概率矩阵，利用候选关键词的角余弦位距作为权重转移概率矩阵中的元素的值，候选关键词的角余弦位距 s_{ij} 计算如式（5）所示。

$$s_{ij}=1-\frac{w_{v_i}\bullet w_{v_j}}{\|w_{v_i}\|\|w_{v_j}\|} \quad (5)$$

其中：w_{v_i}、w_{v_j} 为有相连边的两个顶点各自的词向量，该值是利用 FastText 利用数据训练得到的。但是，在多次实验过程中，会出现无法收敛的结果，经过分析文献[14]，是因为在连通图结构中，如果有顶点的出度为 0，在经过有限次迭代过程后，所有的顶点的值将变成 0,这被称为“等级泄漏”。经过实验设计，为避免这一问题，将上式中的概率转移矩阵进行调整，将每一元素加上所在列所有元素和 $sum(E_i)$ 所得的值作为每一元素的最终值。则上式转变如下：

$$s_{ij}=1-\frac{w_{v_i}\bullet w_{v_j}}{\|w_{v_i}\|\|w_{v_j}\|}+sum(E_i) \quad (6)$$

最终改进的权重转移概率矩阵 M 如下：

$$M=\begin{pmatrix} s_{11} & \cdots & s_{1n} \\ \vdots & s_{ij} & \vdots \\ s_{n1} & \cdots & s_{nn} \end{pmatrix} \quad (7)$$

最终的矩阵运算公式如下：

$$WS(V_i)=(1-d)+d*M*WS(V_j) \quad (8)$$

利用以上的式（8）设计实验，迭代次数的上限值 $T=100$，收敛误差为 0.0001，最终提取 *TopK* 个词语为该文档的关键词。

4 实验结果与分析

4.1 实验数据及评价标准

本文使用来自搜狗实验室的全网新闻数据共 1.4 GB 作为 FastText 的训练集，数据包含了来自若干新闻站点 2012 年 6 月—7 月期间国内，国际，体育，社会，娱乐等 18 个频道的新闻数据。随机抽取字数在 500 以上的新闻内容作为测试集合，共计 70 篇。在内存为 16 GB，系统为 Ubuntu16.04LTS 的计算机上，训练 FastText 词向量模型用时两小时，获得词向量模型文件大小为 3.8 GB。针对测试集，采用多人人工交叉标注的形式提取新闻关键词，每篇新闻人工提取 10 个关键词作为人工标注的结果集（通常 10 个关键词足以概括一篇新闻主要内容）。同时对于测试集，基于 FastText 结合 TextRank 算法进行关键词抽取。

除此之外，基于相同的测试集，采用传统的 TF-IDF、

TextRank、FastText 结合 TextRank 模型结果做交叉对比。TF-IDF 以及 TextRank 算法按照 Python 开源第三方工具进行验证，同时在源码基础上进行优化实现 FastText 的融合。按照信息检索中的精确率 P 、召回率 R 以及 F 值进行统计对比，三种指标计算公式如下：

$$P=\frac{N(人工标注集合\cap抽取集合)}{N(抽取集合)} \tag{9}$$

$$R=\frac{N(人工标注集合\cap抽取集合)}{N(人工标注集合)} \tag{10}$$

$$F=\frac{2*P*R}{P+R} \tag{11}$$

4.2 实验结果

在此次实验中，有两个参数影响着 TextRank 算法和 FT-TextRank（本文提出的算法，FastText-TextRank，以下简称 FT-TextRank）实验的结果，一个是关键词个数 key，另一个是共现窗口大小 window，初始设置为 w=8，而 TF-IDF 算法是属于传统统计算法，算法实现无这一参数，本文利用控制变量的原则，进行了相关实验。以下表 1、图 4 中的结果均是在 w=8 下的实验完成的。

表 1 实验抽取结果

抽取数/个	算法	P/%	R/%	F/%
5	TF-IDF	51.42	25.72	34.29
	TextRank	53.71	26.86	35.81
	FT-TextRank	54.86	27.43	36.57
7	TF-IDF	44.08	30.85	36.30
	TextRank	48.76	34.14	40.16
	FT-TextRank	51.43	36.00	42.35
10	TF-IDF	36.42	36.57	36.49
	TextRank	41.43	41.43	41.43
	FT-TextRank	47.29	47.29	47.29

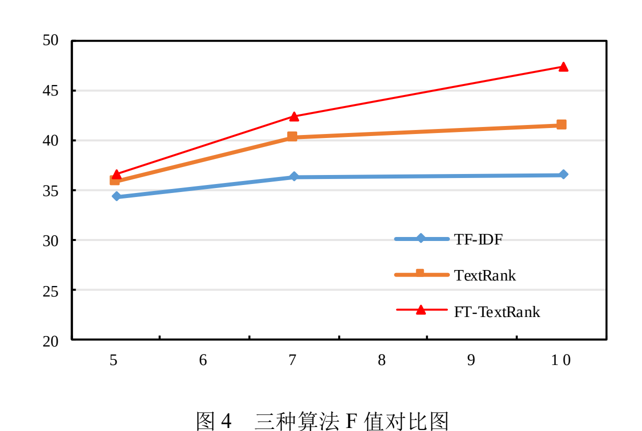


图 4 三种算法 F 值对比图

根据以上实验结果，可以看出随着抽取关键词的个数的增加，FT-TextRank 算法相对 TF-IDF 算法和 TextRank 算法提升效果更加明显，本文提出方法的 P 值、R 值以及 F 值均高于另外两种方法。这也直接的证明了本文提出的方法优于传统 TF-IDF 算法和 TextRank 算法。

此外，对实验结果产生明显影响的另一参数是共现窗口 window，共现窗口大小决定了权重转移概率矩阵的稠密，从而影响抽取结果，同时设置抽取关键词个数 k=10 的情况下，以下图 5 是不同共现窗口大小 w 值下 TextRank 算法和 FT-TextRank 算法的抽取结果 F 值的对比图。

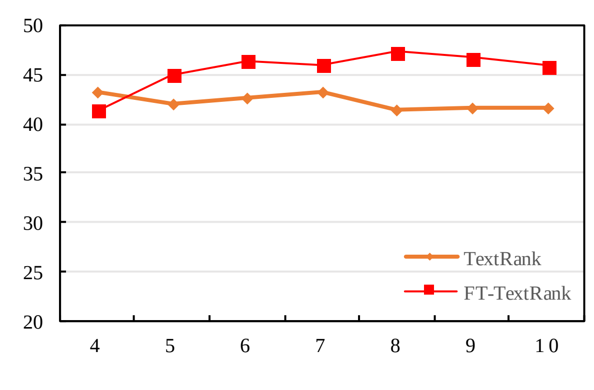


图 5 不同 w 值抽取结果 F 值

图 5 可以看出随着 w 值的增加，TextRank 算法的抽取结果 F 值在降低，而本文提出的 FT-TextRank 算法 F 值在增加，在 w=8 时，效果最好。

5 结束语

一篇文档的关键词是该文档主题内容的直接反映，所以算法提取关键词的结果需要能相对准确地体现出文章的主题内容。但是文字是高度抽象的符号，是人类特有的属性，包含有丰富的语义，所以需要翻译成机器容易理解的表达。因为 FastText 工具优异的性能，同时能得到更好的词向量表征，所以基于隐含主题分布的思想和利用词语的语义性差异能提升关键词抽取的效果。

实验结果表明，本文提出的改进方法，能够提升结果的准确性。接下来的工作是考虑优化词向量模型，使得词向量能包含更丰富的语义特征来进一步提高关键词抽取的效果。同时，因为目前还没有标准的测试集，而且考虑到语义的相似性，准备改进实验结果的评价方法，结合词语语义的相似度来对结果的准确度进行优化。

参考文献：

[1] Blei D M, Ng A Y, Jordan M I. Latent Dirichlet allocation [J]. Journal of Machine Learning Research, 2003, 3 (1): 993-1022.

[2] Li J, Zhang K. Keyword extraction based on tf//idf for Chinese news document [J]. Wuhan University Journal of Natural Sciences, 2007, 12 (5): 917-921.

[3] Mihalcea R, Tarau P. Textrank: Bringing order into text [C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2004.

[4] 刘俊，邹东升，邢欣来，等. 基于主题特征的关键词抽取 [J]. 计算机应用研究, 2012, 29 (11): 4224-4227.

[5] 罗燕，赵书良，李晓超，等. 基于词频统计的文本关键词提取方法 [J]. 计算机应用, 2016, 36 (3): 718-725.

[6] 耿焕同，蔡庆生，于琨，等. 一种基于词共现图的文档主题词自动抽取方法 [J]. 南京大学学报：自然科学版, 2006, 42 (2): 156-162.

[7] 顾益军，夏天. 融合 LDA 与 TextRank 的关键词抽取研究 [J]. 现代图书情报技术, 2014, 30 (7): 41-47.

[8] 夏天. 词向量聚类加权 TextRank 的关键词抽取 [J]. 数据分析与知识发现, 2017, 1 (2): 28-34.

[9] 李鹏，王斌，石志伟，等. Tag-TextRank：一种基于 Tag 的网页关键词抽取方法 [J]. 计算机研究与发展, 2012, 49 (11): 2344-2351.

[10] 李跃鹏，金翠，及俊川. 基于 word2vec 的关键词提取算法 [J]. 科研信息化技术与应用, 2015, 6 (4): 54-59.

[11] 姜芳，李国和，岳翔. 基于语义的文档关键词提取方法 [J]. 计算机应用研究, 2015, 32 (1): 142-145.

[12] Ortega F J, Troyano J A, Galán F J, *et al.* Str: A graph-based tagging technique [J]. International Journal on Artificial Intelligence Tools, 2011, 20 (5): 955-967.

[13] Cruz F, Troyano J A, Enríquez F. Supervised textrank [C]// Proc of the 5th International Conference on Advances in Natural Language Processing. Berlin: springer, 2006: 632-639.

[14] Brin S, Page L. Reprint of: The anatomy of a large-scale hypertextual web search engine [J]. Computer Networks, 2012, 56 (18): 3825-3833.

[15] Bojanowski P, Grave E, Joulin A, *et al.* Enriching word vectors with subword information [J]. arXiv preprint arXiv: 1607. 04606, 2016.

[16] Joulin A, Grave E, Bojanowski P, *et al.* Bag of tricks for efficient text classification [J]. arXiv preprint arXiv: 1607. 01759, 2016.